Case Study
Optimization of Technical Systems

Kumpal Khokhariya - 11266110 Navya Thirakala - 11273787 Venkatasumadhar Pabolu -11273758

Abstract

This study explores the application of machine learning to predict maximum amplitude vibrations in turbocharger systems, reducing reliance on costly strain gauge measurements and simulations. Turbochargers experience internal vibrations influenced by excitation order, mode shapes, and nodal diameters. Traditional high-fidelity simulations which is more accurate but computationally expensive for real-time applications. To address this, we evaluate various machine learning models and deep learning architectures. Our analysis reveals that while neural networks require large datasets and extensive computational resources, XGBoost-SVR hybrid modelling provides a more efficient and accurate solution for independent structured data. By leveraging XGBoost for primary trend detection and SVR for refining residuals, we achieve improved prediction accuracy with a significantly lower Mean Squared Error (MSE). The study highlights the effectiveness of hybrid AI approaches in engineering applications, demonstrating their potential to optimize turbocharger performance while reducing computational costs and experimental limitations.

1. Introduction

Turbochargers improve engine efficiency by increasing air intake, but they also experience internal vibrations due to excitation forces and structural movements. If not controlled, these vibrations can lead to mechanical failure. Traditionally, strain gauges are used to measure vibrations and take corrective actions, but this approach is both expensive and time-consuming.(1)

This case study explores the use of machine learning to predict maximum vibration amplitudes, reducing reliance on strain gauges. The dataset includes key operational and design parameters such as excitation order, mode shapes, and nodal diameters, which significantly impact vibration patterns (Figure 1). Various machine learning models, including XGBoost, Random Forest (RF), Support Vector Regression (SVR), and deep learning architectures like NNs, RNNs, LSTMs, and GRUs, are tested to understand their effectiveness. Each model is evaluated to determine the most accurate and efficient approach for this prediction task.

Recurrent models like LSTMs and GRUs excel at capturing sequential and time-dependent patterns, while regression-based models like XGBoost and SVR perform well with structured tabular data. Since vibration data varies across different operating conditions, selecting the right model is crucial. To improve performance, exploratory data analysis and feature engineering are used to preprocess the data and identify key patterns. By replacing traditional simulations with machine learning predictions, this study highlights a cost-effective alternative to extensive strain gauge installations in turbocharger systems. The results showcase how machine learning is reshaping traditional engineering processes.

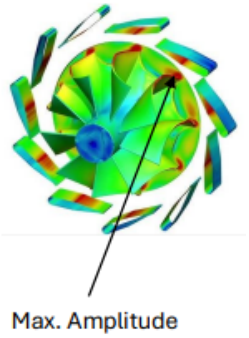

Figure 1. Visualization of vibration patterns influenced by excitation order, mode shapes, and nodal diameters.
(2)

2. Technical Background

2.1. Excitation Order (*I*)

Excitation Order (I) represents the excitation frequencies proportional to rotational speed (RPM). It defines specific vibration patterns affecting the turbocharger system. Higher excitation orders often amplify vibrations at specific fre-

quencies, which can be expressed as:

$$J = I \times \frac{H}{60}, \tag{1}$$

where J is the excitation frequency in Hz and H is the speed in RPM. The amplification of vibrations caused by resonance with specific orders makes this feature critical for predicting the Nominal Amplified Amplitude (N).

2.2. Mode (L)

Mode (L) corresponds to the natural modes of vibration, or eigenfrequencies, of the system. These modes depend on the geometry, mass distribution, and material properties of the turbocharger. When excitation frequencies align with the eigenfrequency of a mode, significant resonance occurs, leading to amplified vibrations. This relationship is expressed as:

$$J \approx f_{\text{natural}}(L), \tag{2}$$

where $f_{\text{natural}}(L)$ represents the eigenfrequency for a specific mode.

2.3. Node Diameter (K)

Node Diameter (K) represents the geometric distribution of nodal points where no vibration occurs. It influences how vibration energy propagates through the system. Variations in K affect the stress distribution, altering vibration patterns and amplitudes.(3)

As per above technical background, we can say that the I, L and K is the important feature to analyze vibration patterns, Also, we examine the relationship between excitation frequency and amplitude by plotting J against N to observe how different frequencies amplify vibrations.

3. Exploratory Data Analysis

3.1. Identifying and Encoding Categorical data

In order to include every feature and make the machine learning models understand the data we encode the data.In given dataset we have two categorical features C (Measurement Series) and F (Strain Gauge) are encoded.We follow the unique encoding technique involves:

- Extracting the last character of C and mapping it to an ordinal index.
- Assigning unique integer mappings to F values for categorical encoding.

3.2. Correlation Analysis

A correlation matrix is extracted to find the effect of each column with respect to other . Key findings include:

- Strong correlations between I (Excitation Order) and J (Frequency), reinforcing their linear relationship.
- Moderate correlation between L (Mode) and N (Nominal Amplified Amplitude), suggesting resonance effects at specific modes.
- Nonlinear interactions between K (Node Diameter) and N, highlighting geometric influences on vibration behavior.

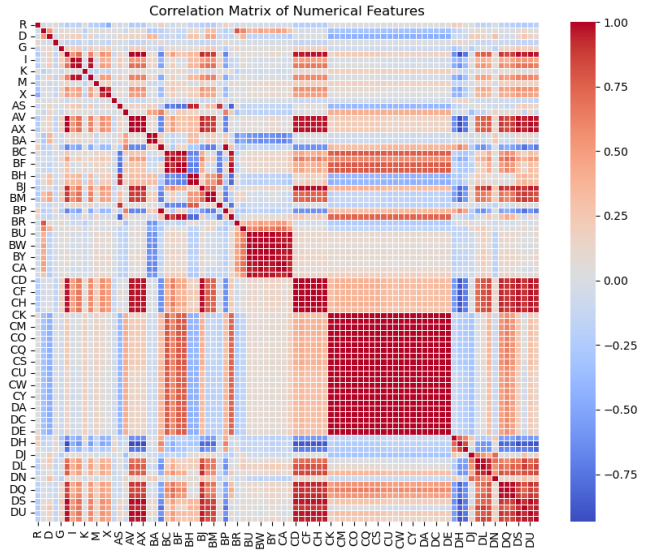

Figure 2. Correlation matrix of key features influencing vibration amplitude (N).

3.3. Grouping and Standardization

Grouping of data is done based on technical analysis and features included here are C (Measurement Series), F (Strain Gauge), I (Excitation Order), K (Node Diameter), and L (Mode). Groups with fewer than five data points are discarded to maintain statistical validity. This method can avoid deviation the model performance.

Unlike deep learning models, XGBoost does not require strict normalization, but we apply standardization to ensure numerical stability.

3.4. Feature Correlation with Target Variable

Correlation between input features and the output variable (target variable) is crucial in understanding relationships in data.With the technical background and requirements we understand that the amplitude is the required output to be predicted.The correlation of individual features with N

(Nominal Amplified Amplitude) is examined to understand their impact on vibration characteristics.

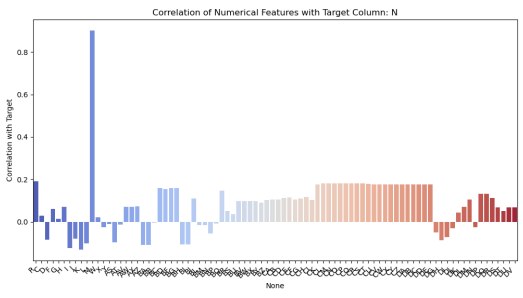

Figure 3. Correlation of various features with the target variable *N*.

3.5. Insights and Key Findings

- Feature encoding and grouping improve data consistency and structure.
- Unlike deep learning, XGBoost does not require strict normalization but benefits from well-distributed features.
- Feature importance analysis shows that *I* (Excitation Order) and *L* (Mode) significantly contribute to predicting *N*.
- Correlation analysis reveals linear and nonlinear dependencies affecting vibration amplitudes.

These insights will guide hyperparameter tuning and model optimization for XGBoost in subsequent sections.

4. Feature Engineering

During our data analysis, we leveraged both statistical insights and domain knowledge of the turbine system to perform feature engineering and reduce the total number of features in the dataset. One of the key techniques we applied was computing the mean of several similar features and incorporating this aggregated value as a new feature. Subsequently, the original individual features were removed to simplify the dataset while preserving the relevant information.

For the first instance, we calculated the mean of features **TvV_1 to TvV_8**, which represent the temperature measurements of valve **V1 to V8**, and introduced this mean as an additional feature. Since all these features exhibited a similar relationship with the target variable, replacing them with their mean helped retain the essential information while reducing dimensionality. This approach enhances model efficiency and mitigates redundancy without compromising predictive performance. Similarly, we applied the same approach to **TnV_1 to TnV_6**, which represent the temperature values of valves **N1 to N6**, and introduced their mean as an additional feature.

We also followed this method for other feature groups. We computed the mean of the **static pressure of the sealing cover** (covering positions **01 to 20**) and added it as a new feature.

Additionally, we averaged the **lube oil pressure** features and the **lube oil temperature** features, introducing their mean values as new features. Finally, we calculated the mean for **axial bearing temperature** and **turbogear bearing temperature**, combining them into another new feature.

By the end of this process, we had introduced **eight new features** and removed the original individual features that contributed to them. This approach not only streamlined the dataset but also maintained the essential patterns, improving model efficiency without sacrificing predictive power.

Below is the image of class define for feature reduction and it's use case.

5. Model Selection Justification: XGBoost vs. Neural Networks

5.1. Performance Comparison

We initially experimented with Neural Networks (NN), including RNN, LSTM, and GRU, for modeling. However, the results showed a high Mean Squared Error (MSE) of around 27.38. The model struggled to capture the actual pattern of the data in the validation set and performed poorly on unseen data, failing to predict values accurately. After further research, we realized that these neural network techniques require a large dataset to learn effectively. As a result, we decided to switch to less complex models such as Random Forest (RF), Decision Trees (DT), and XGBoost. As expected, these models significantly improved performance. With our XGBoost-SVR hybrid model, we achieved a much lower MSE of 9.68 on the unseen dataset.

5.2. Technical Comparison

- Advanced neural network models require large-scale datasets to perform well. In contrast, XGBoost is more effective on smaller datasets. Since our dataset is relatively small for deep learning models, using neural networks led to issues like overfitting and underfitting.
- Neural networks often struggle with structured tabular data, whereas regression-based models tend to perform better in such cases.
- Training deep networks also demands significant computational resources, whereas XGBoost can be effi-

ciently trained with hyperparameters tuning while requiring considerably less memory.

- XGBoost naturally captures feature interactions, whereas neural networks rely on multiple deep layers to learn these relationships. This means that XGBoost can identify and utilize complex patterns in the data without requiring extensive feature engineering. XGBoost provides clear insights into feature importance, allowing us to understand which variables have the most influence on the model's predictions.
- Unlike neural networks, which require explicit imputation of missing values, XGBoost can handle missing data internally. This makes the modeling process more efficient and reduces the risk of errors introduced by imputation methods.
- To prevent overfitting, XGBoost incorporates L1 (Lasso) and L2 (Ridge) regularization techniques. These methods help stabilize the model by penalizing overly complex patterns, making XGBoost more robust and generalizable than deep neural networks, which often require additional techniques to manage overfitting effectively.

5.3. Advantages of XGBoost-SVR Hybrid

- XGBoost is great at capturing the main patterns in the data, but it might miss some of the smaller, more detailed trends. This is where Support Vector Regression (SVR) comes in—it's used to fine-tune the residuals, or the errors left by the initial XGBoost model. By training on these residuals, SVR helps to adjust and correct the model's predictions, improving accuracy without the risk of overfitting. This hybrid method takes advantage of XGBoost's ability to handle important features while using SVR to polish the predictions, making the final model more precise.
- Residual training is useful because it focuses on the discrepancies between the model's current predictions and the actual values. By using SVR on these residuals, the model can learn to correct its mistakes, effectively "refining" the output. It helps to capture those smaller patterns the initial model might have missed, improving the overall performance and making the model more robust. This method enhances the model's ability to generalize, preventing overfitting that could occur if we tried to improve the original model too much.

5.4. Visual Comparison

The figure below compares different model approaches in predicting the value of vibration (N) as a function of frequency (J) for the group MR002bh. The plot contains four curves:

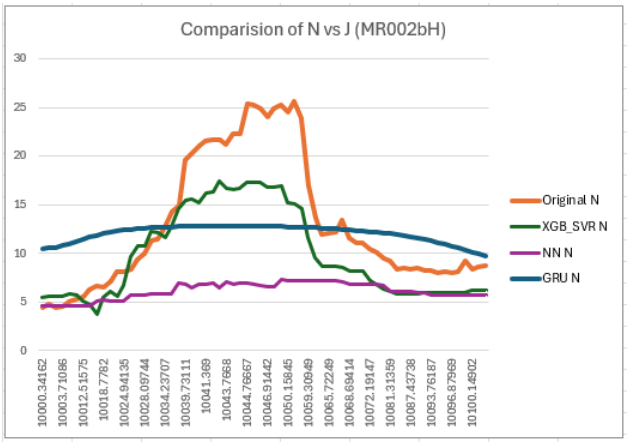

Figure 4. Comparison of Original vs. Predicted Target Values by XGBoost-SVR, NN and GRU

- Original(N) (orange line): Represents the actual recorded vibration values.
- XGB_SVR(N) (Green Line): Represents predictions from the XGBoost-SVR hybrid model, which follows the general trend of the original data but smooths out some variations which give MSE of 9.68.
- NN(N) (Purple Line): Represents predictions from a Neural Network (NN) model, which remains relatively stable and fails to capture the significant fluctuations present in the original data which give MSE of 38.82.
- GRU(N) (Blue Line): Predictions from a Gated Recurrent Unit (GRU) model, which also remains stable and does not fully match the peaks and deviate from the original data which give MSE of 27.38.

The XGB-SVR hybrid model effectively captures the overall trend of the original data, though it slightly underestimates peak values. In contrast, the NN and GRU models struggle to learn the data's variability, producing a relatively flat response that fails to capture sharp fluctuations. This comparison highlights the strength of the XGBoost-SVR hybrid model in handling the dataset's complexity, making it the most accurate among the tested approaches.

6. XGBoost-SVR Hybrid Model

After evaluating multiple models, we chose to proceed with the XGBoost-SVR hybrid approach due to its superior performance and higher prediction accuracy. To ensure optimal results, we first conducted a comprehensive hyperparameter grid search to fine-tune the XGBoost model, selecting the best combination of parameters for improved efficiency. Once the XGBoost model was trained, we calculated the

residuals from its predictions on the training dataset. These residuals represent the model's error, which we then used as training data for a Support Vector Regression (SVR) model. The SVR model was specifically trained to predict these residuals, allowing us to refine the overall predictions. In the final step, we combined the original predictions from XGBoost with the SVR-predicted residuals, effectively correcting errors and enhancing the model's accuracy. This hybrid approach leverages the strengths of both models—XGBoost's ability to capture key patterns in the data and SVR's fine-tuning capabilities—resulting in a more precise and reliable predictive model.(4)(5)

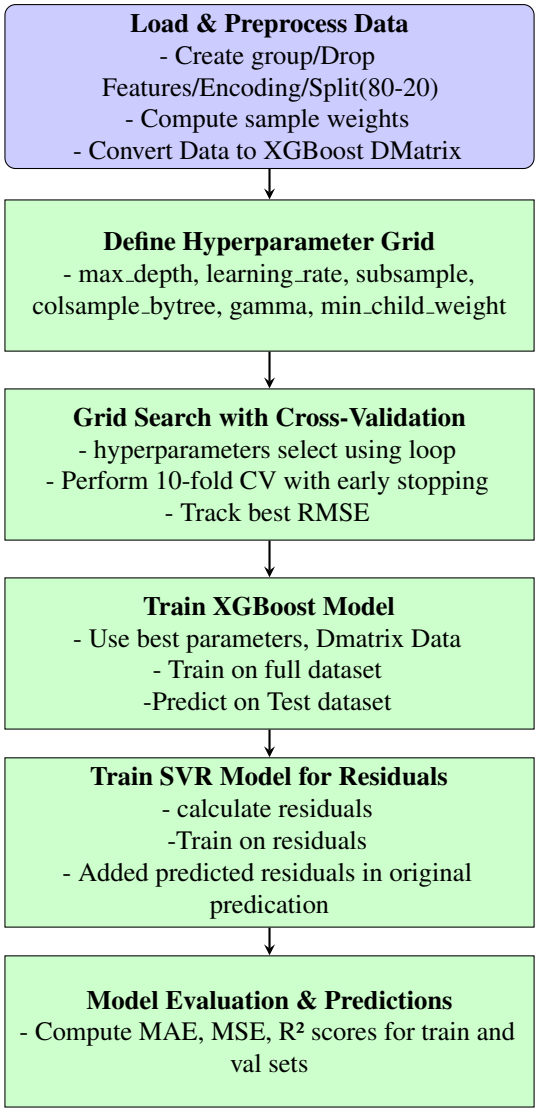

Figure 5. Block diagram of XGBoost model training process.

7. Observations and Conclusion

After conducting the experiment, we observed that the XGBoost-SVR hybrid model performed well in predicting actual values for the validation dataset, as shown in Figure 6.

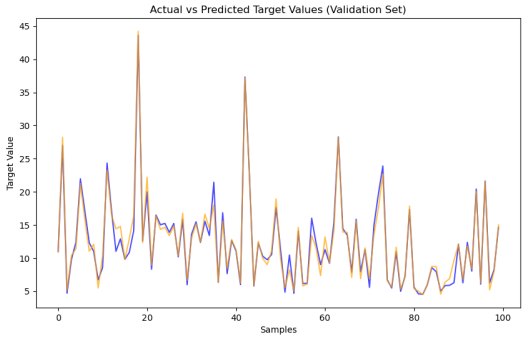

Figure 6. Actual vs Predicted Target Values

However, when tested on a completely unseen dataset—where the target variable was not provided, the model attempted to predict the actual peaks in the data but was not as accurate as it was on the validation set, as shown in Figure 7. Despite this, the distribution of

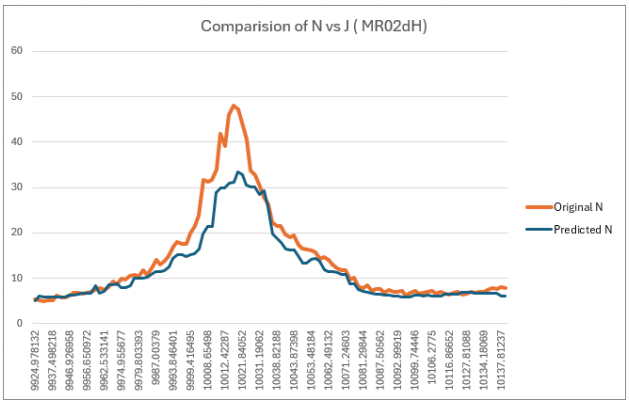

Figure 7. Actual vs Predicted MR02dH

predicted values closely resembled the distribution of actual values, as illustrated in Figure 8.

We believe this deviation in predictions for the unseen dataset is primarily due to the large gaps between successive target values. While other features in the dataset do not exhibit such significant variations, there are two possible explanations for this behavior. First, the dataset may be missing a few critical features, or some rows may have been inadvertently removed, leading to inconsistencies in

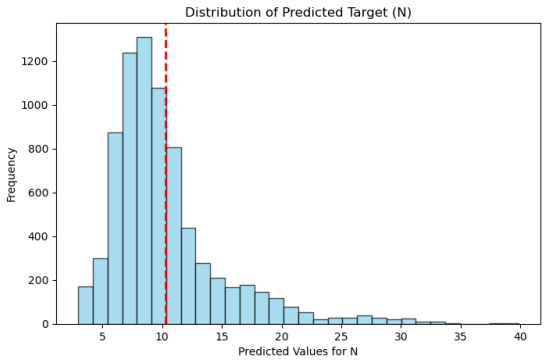

Figure 8. Distribution of Predicted Target (N)

the target variable. Second, external factors affecting the simulation setup could be influencing the target variable, causing these large deviations between consecutive data points.

8. Further Developments

There are several potential methods we can explore to further improve our model. For instance, Physics-Informed Neural Networks (PINNs) could be useful when dealing with limited data and complex relationships between features, but this requires a clear understanding of the key feature interactions. Additionally, we can refine our approach through feature engineering, leveraging domain knowledge to extract more meaningful insights from the data. Another promising direction is the development of hybrid models that combine simulation techniques with AI, potentially enhancing predictive accuracy and robustness.

References

[1] J. Allport, "Turbocharger blade vibration: Measurement and validation through laser tip-timing," *ResearchGate*, 2014. [Online]. Available: https://www.researchgate.net/publication/260405273_Turbocharger_blade_vibration_Measurement_and_validation_through_laser_tip-timing

[2] N. C. Puetz and M. C. G. 3, "Case study," Online, 2024, accessed: February 2025. [Online]. Available: https://git-ce.rwth-aachen.de/noah_christoph.puetz/man-cs_group3/-/blob/main/Case_Study.pdf?ref_type=heads

[3] T. Bartz-Beielstein, "Hcf features," Presentation slides, 2024, accessed: February 2025. [Online]. Available: https://git-ce.rwth-aachen.de/spotseven-lab/cs-10-man-ws-24-25/-/blob/main/Documents.d/hcf_features.pptx?ref_type=heads

[4] T. Chen and X. Developers, *XGBoost Documentation*, Online, 2024, accessed: February 2025. [Online]. Available: https://xgboost.readthedocs.io/en/stable/

[5] S. learn Developers, *sklearn.svm.SVR - Support Vector Regression*, Online, 2024, accessed: February 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html